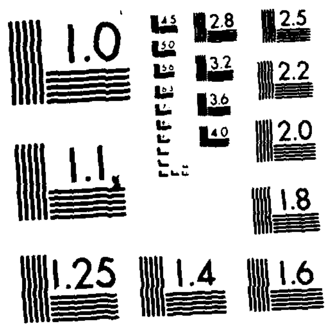
AD-A138 840 BEYOND THE DATA BASE: TECHNOLOGY FOR INFORMATION RESOURCE MANAGEMENT(U) ROME AIR DEVELOPMENT CENTER GRIFFISS AFB NY P M LANGENDORF DEC 83 RADC-TR-83-312 1/1

UNCLASSIFIED F/G 9/2 NL

END
DATE
FILMED
4-84
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

ADA138840

RADC-TR-83-312
In-House Report
December 1983

BEYOND THE DATA BASE: TECHNOLOGY FOR INFORMATION RESOURCE MANAGEMENT

Patricia M. Langendorf

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-83-312 has been reviewed and is approved for publication.

APPROVED: John R. Bratge

JOHN R. BRATGE
Chief, Intelligence Data Handling Branch
Intelligence & Reconnaissance Division

APPROVED: Albert A. Jamberdino

ALBERT A. JAMBERDINO
Acting Technical Director
Intelligence & Reconnaissance Division

FOR THE COMMANDER: John A. Ritz

JOHN A. RITZ
Acting Chief, Plans Office

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-83-312	2. GOVT ACCESSION NO. AD-A138940	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) BEYOND THE DATA BASE: TECHNOLOGY FOR INFORMATION RESOURCE MANAGEMENT		5. TYPE OF REPORT & PERIOD COVERED In-House Report 6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) Patricia M. Langendorf		8. CONTRACT OR GRANT NUMBER(s) N/A
9. PERFORMING ORGANIZATION NAME AND ADDRESS Rome Air Development Center (IRDA) Griffiss AFB NY 13441		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 459416SB
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (IRDA) Griffiss AFB NY 13441		12. REPORT DATE December 1983 13. NUMBER OF PAGES 44
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

data base management systems (DBMS)
computer-based data management
information resource management system development

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report is about data in our society. It first discusses values, how people feel about using data. It then discusses how man has used the technology that is available as tools in handling data, and how data handling has changed as technology changes. Information resource management is now changing very rapidly as new technology becomes available.

The report argues that the complex programs we call data base management systems (DBMS) are an artifact of monolithic computers with hierarchies of

secondary storage managed by complex general purpose operating systems. The next step will take the DBMS apart, distributing the data management functions to provide the same capabilities with less constraints.

The report concludes with a discussion of theory regarding computer based data management. It covers the intellectual ferment following the introduction of random access secondary storage, the development of CODASYL, and the relational model. It briefly discusses the familiar arguments regarding the need to separate logical data definition from the physical location of data, stressing different representational requirements. The report concludes with the author's view of what is needed to form an adequate foundation for information resource management system development.

The computerized data system will have the same impact upon society that the printing press had, and learned people are reacting as learned people reacted when printing was developed. They are arguing that the new technology will enslave the common man and should be suppressed. In a sense both are right. The only possible defense against the printing press is to learn to read. We succeeded that time. Print fonts and spelling were standardized so everyone could learn to read, and then we insisted that everyone learn to read. Now we have to standardize the interface to computerized data systems and teach everyone to use them. A person who is illiterate in computerized data handling in the future world will be as defenseless as the person who cannot read is defenseless now.

Andrei Ershov, Soviet Academician
paraphrased from address, Spring
Joint Computer Conference, 1967

This paper is about data in our society. It first discusses values, how people feel about using data. Then it discusses how man has used the technology that is available as tools in handling data, and how data handling has changed as technology changes. Information resource management is now changing very rapidly as new technology becomes available.

The paper argues that the complex programs we call data base management systems (DBMS) are an artifact of monolithic computers with hierarchies of secondary storage managed by complex general purpose operating systems. The next step will take the DBMS apart, distributing the data management functions to provide the same capabilities with less constraints.

The paper concludes with a discussion of theory regarding computer-based data management. It covers the intellectual ferment following the introduction of random access secondary storage, the development of CODASYL, and the relational model. It briefly discusses the familiar arguments regarding the need to separate logical data definition from the physical location of data, stressing different representational

requirements. The paper concludes with the author's view of what is needed to form an adequate foundation for information resource management system development.

VALUES PAST

Yes, Hans Olderwick was my brother, and so was Andrew Olafson. We didn't all use the same last name in Norway. If a family all used the same last name, the government could keep records, and it would use those records against the family. It would draft men for the army. It would use them to find out how much tax the family could afford.

Conversation with my grandfather, Ivor Olson
circa 1945

Traditionally, most cultures had a decided cultural bias toward limiting data collection, based upon an experienced distrust of people who wanted data. Much of the opposition to the introduction of social security in this country was based upon the argument that it would enable the government to keep accurate records. My father carefully asked the lumberjacks who worked for him what name they wanted them to give social security. He didn't ask their name since that was agreed to be none of his business.

Even how-to information was controlled and access to information required membership in organizations owning the information. Crafts from ironworkers to miners to bridgebuilders were passed on to initiates and others were carefully excluded. Pythagoras set the penalty for teaching his geometry to unauthorized people at death, and rigorously enforced it.

The development of the modern world was based in large part upon learning how to obtain and use information. Trade secrets were captured, codified and taught. Patent systems were introduced to encourage inventors to share the secret details of their inventions. Formal procedures for managing data were developed. The library and the filing cabinet were augmented by the computerized repository. With facts available to support theorizing, technology exploded. The scientific method formalized a role for experimentation, collecting and

retaining data to prove or disprove hypotheses. And as we did these things, our attitudes changed.

We are in agreement that how-to knowledge should be formalized and taught, including agreement that access to teaching should not be arbitrarily limited.

Our attitudes towards data are in a state of flux* as witness the same congress passing one bill guaranteeing privacy and another access to government information without serious debate on the unavoidable conflict between these two goals.

We are developing an appreciation for data as a major property right of organizations.

We appear to have reached a consensus that capturing data is legitimate unless other laws must be violated to capture it, with the resulting data belonging to the people who capture it. We are also reaching a consensus that if access to data is offered to anyone, then objective standards for access to data should be developed and enforced. In other words, that the standard commercial laws apply to data. More and more, access to collections of data are being offered in return for money.

VALUES PROBLEMS

I ask you to really believe that the computer was developed thirty years earlier than it was* and the state of the computer art in 1936 was what it is now. Now--remember what Hitler did to the Jews? Without a computer, could he have done it?

Professor Joe Weisenbaum, MIT
personal conversation, Aug 1966

Almost fifty years after his demise, it's hard to remember that Hitler didn't have computers. Using only manual records, he defined the

term 'Jewish ancestry, defined a Jew as a person with one fourth Jewish ancestry, and then he found over 95% of them, some of whom didn't know their ancestry. This required a massive effort involving major resources. We would be wise to remember that if there is a next time, finding people who meet any arbitrary definition will be easy. Computer-based data will be available somewhere.

A computer has no ethics. It is a tool, doing what it is programmed to do. There is literally nothing a computer can do that determined people willing to pay the price cannot do without it, with the possible exception of real-time control. What the computer does is change the cost and difficulty of doing things, often by many orders of magnitude.

Much of the advantage, or disadvantage, from a computer-based information system is due to the analysis that preceeded it. The most effective approach is to stop just before installation of the computer, and consider what you really want to do.

We have the ability to provide access to accurate data to many people under many conditions. In using this ability, we have learned that many traditional procedures were wisely designed to have data, and limit access to that data by making access difficult.

An example is that the full personnel record of each employee under his/her supervision is available to each supervisor in the US federal service. The supervisor doesn't have to state a reason for wanting to review the file. However, the supervisor must make an appointment and go to another office usually located in another building to read the file. When computers and communication equipment enabled access to personnel files maintained by a personnel office be provided in the supervisor's office, some organizations provided that access. They removed it because employees felt the supervisors were misusing it snooping into their personal affairs without sufficient cause.

The world we live in has fact availability far beyond our understanding of the interrelationships between facts and the meanings that can be inferred from them. We have changed from a world where new information was scarce to one so inundated in data that almost any concievable information can be extracted from data that is routinely captured. We haven't yet learned how to use data or to limit misuse of data.

We haven't learned how to limit access to data, but we are rapidly developing a consensus that data in computerized form belongs to somebody who is responsable for maintaining integrity of the data. The owner of the data is also responsible for authorizing access to data. The usual authorization for access involves paying money, and that money is used to maintain both the data and the organization that owns it.

The most grandiose data availability scheme is Euronet(1), where the European telephone companies offer a service. Anyone who wants to sell access to computerized information can enter the net. They just have to provide a standard gateway. Anyone with appropriate equipment who wants to use it can do so through the phone lines. Euronet will bill the user and pay the providers. In the United States, various data services are available. However, the user must usually sign up to the individual data service, and receives individual bills for usage.

We haven't yet learned how to exploit the data we have. Above all we don't know how to recognize and correlate data that are in fact different aspects of the same subject. This might be called the 'blind man and elephant' problem. Modern analysis is agreed to be concept driven. Technology to go from undigested facts to knowledge of interest demands a model relating the facts to the knowledge. Model building is

just as difficult as it has always been. However, model validation is much easier. We have data now, so we can check the validity of our models if we choose to do so.

> To get the facts first is impossible. There are no facts unless one has a criterion of relevance. Events by themselves are not facts....
>
> The only rigorous method, the only one that enables us to test an opinion against reality, is based on the clear recognition that opinions come first--and that is the way it should be. Then no one can fail to see that we start out with untested hypotheses--in decision making as in science the only starting point. We know what to do with hypotheses--one does not argue with them; one tests them.
>
> Peter Drucker, The effective Executive 1967

We now have the data and processing power to validate individual models, and are finding out that the traditional concepts 'everybody knew' are either gross over-simplifications, or else simply wrong.

> For example, the people who built the first computer-aided language translators found that the grammar they had been taught was both inadequate and wrong. As an instance, English adjectives do have order, and changing order changes meaning.

Wrong concepts lead to wrong decisions. Traditionally this didn't matter much, since decisions had so little impact. The impact of decisions depends upon the ability to carry them out. The better the technology available to carry out decisions, the more destructive wrong concepts can be. Our technology is now very powerful. We urgently need much better tools to adapt concepts to conform to the evidence at hand.

> If witness is needed, the behavior of economic systems managed by Keynsian or Marxist models over-proves the point. Nations have literally destroyed their economies by actions taken in accordance with one of these models. Yet each of these models absolutely demands both a static world and a world where information is free and always available. The decision makers who have ruined their own fortunes as well as their nation must have known the world isn't static and struggled to obtain the information they needed to do their jobs. Yet they took the actions recommended by the model they lived by, and then constructed ingenuous explanations after each failure of their system to behave as their model predicted. And continued to use the same model.

We urgently need to develop the judgement to both question and use computer-based predictions. "The computer says" is sometimes used as a final argument, and that is absurd. The implications behind 'the computer says' are that some model has been programmed into the computer, data has been fed into the model, and the computer has 'made' a prediction. The prediction is as good as the model behind it, which can be very good indeed if the model has been validated, and the assumptions upon which it is based have not been changed. It can also be nonsense.

Sometimes the 'model' hasn't been stated, let alone validated in a defined domain. One of the successful techniques of Artificial Intelligence is to model the behavior of selected experts, to develop what is called an expert system. Properly constructed and used, expert systems can provide a useful aid to decision making. However, there is much danger in using the system indiscriminately without either a sound knowledge base or the judgement of the expert being modelled. We face a real danger in accepting computer predictions and recommendations without adequate consideration of the limitations inherent in the technology upon which they are based.

As we move into the information age, we need methodology to manage concepts in the plural, to track the multitudinous possible interpretations of data and to continually ask the qustion , 'What would disprove this concept?'. We need methodology that we trust enough that when a concept is proven incorrect we accept the proof and use it to develop better concepts. The people who use information well will have the same transient advantage the London bankers had when they knew of Waterloo the day before their competitors. Their colleagues who use

facts to validate concepts will have the more enduring advantages the Japanese have shown in adjusting their economy to change.

VALUES FUTURE

> Information is the name for the content of what is exchanged with the outer world as we adjust to it, and make our adjustments felt upon it. The process of receiving and of using information is the process of our adjusting to the contingencies of the outer environment and of our living effectively within that environment......
>
> It becomes plausible that information ...belongs among the great concepts of science such as matter, energy and electric charge. Our adjustment to the world around us depends upon the informational windows that our senses provide.
>
> Norbert Weiner
> Philosophy of Science, 1943

The ability to validate concepts won't be enough for the world that is on the horizon.

As we move into the information age, we need methodology to manage concepts in the plural, to track the multitudinous possible interpretations of data on the one hand, and to continually ask the contradictory question, 'what would disprove this concept' on the other.

We need methodology that we trust enough that when a concept is proven incorrect we accept the proof. Then we need methodology to deal with the disproven concepts. Sometimes we will be able to use the proof that our concept is incorrect to develop better concepts. Sometimes we will only know that what we thought was correct is wrong, without enough knowledge to identify what is right. This is the most difficult of all intellectual situations. We have to be able to accept the fact that we don't know. We must also find a way to describe what is happening well enough to bring brains to bear on the newly identified missing concept.

The important point is that we must be to able apply the scientific method to the concepts from which we infer meaning from data. However,

technology is of no value unless it is used. We need to do what we know. When a theory is proven or disproven, and our previous beliefs are shown to be in error, we must develop the cultural capability to accept the fact. Even when we don't have another theory to replace the one proven to be in error.

TECHNOLOGY PAST

In the design of data base management facilities, particularly the self-contained functions and the host language data manipulation facilities, there is always some underlying philosophy which governs the compromises made in both design and implementation of the system, and therefore provides a basis for understanding the system.

Feature Analysis of Generalized Database Management Systems, CODASYL Systems Committee Technical Report May 1971

The earliest machines that could reasonably be called digital computers were designed to support census data analysis. Data engines preceeded arithmetic engines. However, in the beginning, data was managed for a known set of reasons, with one specific viewpoint. In fact, most data structures were embedded into procedural languages such as COBOL and PL/1.

We often forget the impact of technology. As long as tape was the only available secondary storage, data aggregates of any size could only be accessed serially and computerized data management was management of files. Consequently, additional uses for data involved constructing new files to support access to the data. The new capabilities available when secondary storage was developed that would support random access to data provided a great increase in capability, coupled with enormous additional complexity.

The DBMS approach is an outgrowth of recognizing that data collected for one purpose could be used for many other purposes as well, and should be collected and managed as a resource of the enterprise as a whole. This significant insight coupled with the development of hierarchies of secondary storage led to development of complex programs to support management of the enterprise database.

A number of commercial software packages have been developed to perform data management. Of course, these DBMS assume the computing environment in which they were developed. That is an essentially Von Newmann machine with an operating system to manage the resources of the system including all storage and the necessity to provide all other data usage support inside the DBMS.

The data base management system as an integrated package of software enabled the computer to be used to manage data as a resource. This was a step increase in capability to handle data, paid for with complex software, and the necessity to define an overall worldview (schema). The DBMS approach assumes a data administration function, where the schema and subschema (views) are defined. "They assume the data administrator has all the smarts, and the rest of the users are all chimpanzees" to quote a disgruntled colleague.(2) While this colorful phrase overstates for emphasis, it makes a pertinent point. The data administrator is responsible for maintenance of the enterprise world view, as well as the consistency and structure of the data under administration.

In implementing a particular data management system, an explicit data model is desirable, and separate semantic and syntactic models of proposed DBMS implementations usually lead to simpler logic and more understanding. Interactions between the two models can be readily documented, leading to explicit documentation of both representation-independent problem statements and the implementation decisions. Unfortunately, currently available data models cannot deal with more than a single overall perspective. To go farther will require advances in knowledge representation.

Data models are usually referred to as hierarchical, where data items can have only one 'root', network where data items can have two or more 'roots' and relational where data is represented as relations. Both network and hierarchical data models are based mathematically upon set theory and the relational model is based upon first order predicate calculus. These models have been proven in one-to-one correspondence with one another in the sense that each can carry the same information,c.f.(3) and the particular model to be used in a particular application is primarily a matter of preference. However, the appropriate model for the individual application and individuals using the model can greatly increase understanding of the underlying processes. The hierarchical model was developed first, the network second, and the relational last. Each had a decided advantage over it's predecessors in certain applications. Each was strongly promoted after it was developed. In fact, the relational model developed a following with many characteristics of a cult, and recent advances in data models are referred to as post-relational. Post-relational models combine the characteristics of the hierarchical and relational.

As is usual when step increases are achieved, they are highly valued for a time after which they are taken for granted as a utility. The proper place for utilities in computer science is in the operating system, and data management functions are beginning to migrate to the operating system. The first such function was management of input and output to the computer.

The DBMS provided the capability to use all the data of the enterprise as a coherent whole. This capability to use data inside the DBMS as a resource leads naturally to the desire to combine that data

with other data not in the DBMS. Most modern DBMS packages include the capability to access other files resident in the same machine by declaring the file to the DBMS. However, the situation becomes more complex when the data to be accessed resides in another DBMS, particularly another different DBMS hosted by a different machine.

STANDARDS

> Perhaps the biggest problem in the industry today is that ... approaches to data management are not only incompatible with each other, but also are often data incompatible....
>
> Feature Analysis of Generalized Database Management Systems, CODASYL Systems Committee Technical Report May 1971

The initial response to the desire to use data from different DBMS together was to try to develop standards. In the simplest case, such as the Euronet discussed above, where data is simply retrieved from different sources and moved to another for processing, this has worked quite effectively. The designers of Euronet declared an interface that a DBMS must have to enter the net, and vendors interested in selling data through Euronet built it. In fact, most of them changed their DBMS to make the Euronet interface a standard access procedure instead of building the interface. The Euronet experience suggests that many users desiring data from outside their enterprise are satisfied by a network of accessible data bases. They are willing to identify the appropriate data, retrieve it, and move it into their own DBMS for further processing.

Standards are necessary to enable things from different sources to work together, and useful standards are urgently needed in data management. Formal standards activities are in progress to facilitate the development and exploitation of DBMS functions. The ANSI/SPARC DBMS model has 36 components and 44 interfaces designed to explicate the

functions of DBMS, and the standards necessary to provide integration.(4) ANSI Standards Group X3H4, Information Resource Dictionary Systems is the first formal product of SPARC activity. It is in the process of setting interface and content standards. The national Bureau of Standards has also recently published specifications for a standard data dictionary system.(5) Data dictionaries are being offered for sale with defined interfaces to commercial DBMS, where the data dictionary and DBMS run together. However to date, interface to each DBMS is being separately constructed. No overall standard is yet implemented.

A relational data base task group was chartered in 1979, and has published a feature catalog of systems which is a precursor to standards definition. This will at least resolve a great deal of confusion regarding what is, and what is not, relational.(6)

Attempts at more detailed standards, particularly attempts to standardize on one computer and one DBMS haven't been very successful. There are three primary reasons. First, DBMS and data structures determine how the DBMS behaves. Forcing several functions to use the same DBMS has lead to unacceptable operational inefficiencies. Second, there is always the data that wasn't thought of when the overall system was designed, and that is now resident on a different host with a different DBMS, costing a fortune in time and money to change it but needing to be used together. Third and most important, data held in identical DBMS on identical computers still can't be used together without additional software. What is needed for a single DBMS would be one DBMS managing data in several computers at several nodes. This is still a research subject. However, practical considerations of data ownership and control limit the applicability of the single DBMS-many computers

approach to using data from different computers together. Capabilities to use DBMS together are required.

THE UNIVERSE EXPANDS

We want more.

Sam Gompers at various times 1866-1924

The first idea to facilitate using DBMS together was to implement the query language of one on the other. Ari Shoshani(7), then of SDC, has proven this is in general impossible. However, Robert Frankel(8), then at the Wharton School, developed an interpretive interface called a Functional Query Language (FQL) in which general programs can be written, and which can be built 'on top of' any DBMS. Frankel demonstrated a general capability to put a general 'top' on different DBMS that will allow one program to run against all of them. FQL is used extensively in building software for commercial sale.

The first operational system designed to support access to disparate heterogeneous data bases is the ADAPT capability on the highly classified COINS network at the National Security Agency. However, adding the next data base to ADAPT as originally designed becomes progressively more difficult as the number of DBMS increases. Adding another DBMS to FQL does not introduce additional difficulties, and rebuilding ADAPT on FQL has been proposed to NSA. There is no published data regarding usage of ADAPT. However, ADAPT runs on a PDP11/70 attached to the COINS network and queue buildup is not excessive, leading to the inference that queries involving two or more COINS DBMS are comparatively infrequent.(9)

FQL and similar systems solve part of the problem of using data bases together since data can be retrieved from several heterogeneous

systems using the same access capability. However, an PQL system still requires that the user of the data know which DBMS holds the data to be accessed. L.S. Schneider, then of Martin Marietta corporation, designed and built a relational query compiler that could access distributed heterogeneous data bases without requiring the user to know where the data resided.(10) However, this compiler required extensive computations. Additional research is required to establish how to build an operational system with the capability to access distributed heterogeneous data bases where the user doesn't need to know where the data is located.

The difficult problems in using data from different DBMS arise when data from one DBMS is used to specify the data to be retrieved from another DBMS. This capability would certainly be useful. For example: 'identify the poison from these symptoms and find out where the antidotes can be obtained.' Symptoms of poisons and location of antidotes are almost certainly in different DBMS. However, DBMS can be used effectively together without this capability.

PRACTICAL ISSUES

> The ultimate objective of database systems is to make application development cheaper, faster and more flexible. It is important not to lose sight of this overriding objective as we discuss the complex mechanics of database systems.
>
> James Martin
> Computer Database Organization, 1977

> Does it do what I want it to do?
>
> almost all users everywhere

In practice, getting data into the computer in the desired format, with suitable accuracy checks so applications can be performed almost always takes at least half the operating resources required to operate the DBMS. Most operational DBMS with good reputations were designed to facilitate input. Considering the economics of DBMS, input and data

capture generally have received surprisingly little theoretical attention. This is understandable, since data input and data capture use different technology and require skills and orientation outside computer science. What is not understandable is that DBMS are almost invariably evaluated in terms of the ability to respond to an ad hoc query. The ad hoc query is particularly inapposite in that many DBMS are more often used by programs than by people generating queries.

However systems are evaluated, the practical problems that systems face are usually in getting data into the DBMS in a timely, accurate, and consistent manner.

Data capture requires identifying the data sources, getting authorization if it belongs to someone, and setting up procedures for capturing it. The major advances in input efficiency have come from building gadgets to capture data in the process of doing something else, and building other gadgets to enable comparatively unskilled people to do tedious repetitive work more efficiently. For example, key to disk systems are a major advance over keypunch machines, because they enable data to be directly entered on disk. Whenever possible, well-designed systems build devices to capture data directly in the process of performing another necessary function. For example, automated cash registers capture the data necessary to maintain inventory. With the advent of microcomputers, such automatic data capture is becoming common, generating streams of data that the data owners want managed.

Experience has also established that for many applications, DBMS technology isn't desireable. DBMS software has an overhead that should only be paid when additional usages of the data become important to the enterprise, since data kept in files is much more efficiently managed.

Files can still be accessed to enable the data to be used for other applications if the occasion arises. In particular, if a major portion of a data base must be accessed in a particular order on a routine basis, then the data should be kept in the order in which it will be accessed. Otherwise retrieving the data in that order will saturate the DBMS, crowding out other users. Often, a DBMS is used to facilitate access to data in files, or even outside the computer system.

With data in machine readable form ready to enter the system in ever-increasing quantities, the importance of designing the system to capture this raw data, do whatever transformations are required and enter the data including integrity checks, indexing and building other data access mechanisms in time synchronized with the data arrival speed becomes critical.

The overall system must be designed to operate with human intervention only where human decision-making is required. This again involves detail understanding of the enterprise. The enterprise model is invaluable in designing the data system to act in accordance with a prescribed algorithm.

Of course, sometimes the prescribed algorithm has unrecognized side effects. It is now almost impossible to buy the standard unstylish oxford woman's shoe. This style remained the same for decades, selling enough to justify stocking it to women who preferred substance to style. They are the least expensive product for a shoe manufacturer to make, since they do not involve changing production machinery or working instructions. They are also the least expensive shoe to stock since they will not go out of style. However, the algorithm that was introduced to manage production of women's shoes and apparently sold to all shoe manufacturers, assumed that sales of each style would peak and die off. It is important to stop production to prevent unsalable no-longer-stylish shoes. So a 'stop production when sales no longer increase' order was built into the production manager. Oxford shoes signal stop production, and the production manager stops. To date, the traditional oxford sources have not chosen to override their program.(11)

A great deal of theoretical attention has been paid to preventing problems in concurrency. These problems arise when one person or program is updating a datum or set of related data while another is using it, or when two or more sources are updating the same data. This is solved by appropriate locking of files being updated or by time-stamping to assure that data from different update cycles is not used inadvertently used together. Actually, this problem is not often serious in practice, since data is almost always updated by only one source and little damage is done by locking the domain in which updating is taking place.

A closely related problem can be critical if two or more users expect to use the same item. The military example is where two or more commanders each plan to use the same airplane or shoot the same ammunition. In practice, the resource allocation problem is usually avoided by administrative action of the organization using the data base. The military know who owns the ammunition and the airplane, and an individual who doesn't own it will get authorization from one who does before planning to use it. In airline ticket sales, the transaction takes place when the ticket is actually sold which is defined as deleting it from the data base. The ticket buyer knows that until then the ticket s/he was told about is subject to sale.

Another related problem is data security. It is accepted that if a good system programmer can get into a computer system, they have access to everything in the system. This hasn't been really true under all conditions for several years, but the problem of how to allow individuals access to part of an integrated data base while preventing access to other parts of the same data base does not have an agreed-upon

solution. In practice, starting with a DBMS and schema definition that support identification of data in terms of access authorization, and then using a combination of limiting access to trusted individuals and administrative sanctions provides acceptable security in most situations. Theoretical work has focused upon building provably secure systems. At present it is possible to prove the correctness of only very small fragments of code. Certifiably secure data systems are expected to become available in the late 1980s.

Sometimes conventional DBMS and analytic approaches lead to unacceptable time or resource requirements. This is caused by very complex analytic processes and/or very large data aggregates. The prototype difficult problem is unfolding hills in geological analysis. Several interesting problems remain unsolved. However, successful approaches to these problems have been based upon simulation, notably by Bouille of the Pierre and Marie Curie Institute, Paris France.(12) Another approach which is being exploited to solve the very difficult problems is through artificial intelligence modeling of experts in the subject area.

As mentioned above, the primary user of most DBMS is software, not direct query by people. The requirements for effective interface to programs are different than those for interface to humans. In either case it is important not to throw away knowledge regarding data unintentionally. This point appears to obvious to state, except that in practice standard usage of many DBMS structures do this if used without due care. It is almost impossible to 'normalize' a relational structure without discarding information that must then be laboriously recalculated.

Some theoreticians argue that resources are rapidly decreasing in price while programmers and users are becoming more expensive. Consequently, they argue that the DBMS design should not consider efficiency except in terms of demand upon human users. The problem with this argument is that in practice however many resources have been provided, uses for these resources rapidly exceedes the supply. Upgrading a DBMS system is still a major undertaking. In consequence, how the DBMS interacts with the primary intended use must be considered if the DBMS is to provide effective support.

Computer systems generally behave better if designed by the old Catholic principle of subsidiarity= never do anything at a higher level if it can be done at a lower level. Have the computer perform data selection at the first point where all necessary data is available to minimize data handling. Keep support close to what is being supported, don't let support of individual devices propagate far beyond the device being supported. Keep user support close to the user, printer specific commands close to the printer, and disk directories on the disk itself. Developments on the horizon may well make these practices much easier. However, experience to data suggests that neglect of these practical issues will still lead to unacceptable system performance.

TECHNOLOGY FUTURE

> the main difficulty and challenge of our age is drastic change... no matter how desireable, drastic change is the most difficult and dangerous experience mankind has undergone. We are discovering that broken habits can be more painful and crippling than broken bones...
>
> Eric Heffer, The Temper of our times, 1967

Technological changes are beginning to provide alternative approaches to implementing the capabilities now provided by conventional DBMS. New requirements are arising better served by these alternative

approaches. Streams of technology are converging.

First, microchips have been developed that allow processors to be placed in the disks, printers and other peripherals. This provides a step increase in data handling capability. A data base machine can be fruitfully analysed as a very smart disk. Data base machines are providing a capability to manage flat files very efficiently. These machines have elements of parallelism, and several such machines can be used together as back-end to data management systems. Microchips from which complete data base machines can be tailored are being built to provide a foundation for construction of machines tailored for optimal support to specific applications.

Next, microprocessors with user support software at the work station are beginning to provide the integrated data view previously requiring DBMS software, and DBMS overhead. Work stations managing the subschema appropriate to their applications accessing data in data base machines or even computer-based file systems can be a very adaptable procedure for information resource management, providing DBMS capabilities without DBMS overhead. In fact, for many applications a central processing unit is not necessary. Microprocessor-based work stations support individuals who share a common data base in a data-base machine.

The microprocessor-based logic being introduced into peripheral storage devices enables decisions to be made much earlier in the data retrieval process, limiting the data that must be transferred into core memory, greatly reducing the processing required. Logic in the printer means that only high-level 'how I want it to look' instructions must be passed to the printer, again reducing processing. This logic in

peripherals enables much simpler software packages to perform the DBMS functions.

Most micro-based home computer systems have consistent environments with very smart disks and printers. For example, the Commodore disk drive includes a built-in index on each floppy disk. Very few commercially available disks for larger systems have equivalent capability. Users of these home computers will demand the same consistency and support in their major systems.

Third, the data requirements of Knowledge Management, Active Data Base systems, Artificial Intelligence and Decision Aids require interfaces to data to work effectively and these interfaces are difficult to implement in conjunction with conventional DBMS. Prototype systems are being constructed with LISP machines attached directly to data base machines without involving a general computer.

Finally, application development environments are being built, particularly to support AI applications written in the programming language LISP. This experience establishes that capabilities now provided by integrated DBMS could be provided as mutually consistent sets of utilities managed by the computer operating system in the conventional manner. It also established that the consistent set of utilities would eliminate the overhead and arbitrary constraints of monolithic DBMS. A well-designed programming environment also ensures consistency as new applications are developed.

This would have the added advantage that data management functions would be available in conjunction with the full processing power of the computer. This is important for many forms of analysis. Most conventional DBMS require that data be taken outside the DBMS for

processing, with consequent delays and programming complexity required to perform analysis.

The computer language Ada(13), developed by the DOD for embedded computer functions and to be mandatory for mission-critical software is an example of such an application support environment. Ada is also the first operational language to include real abstract data types. As knowledge regarding how to build computer languages has developed, some computer scientists have argued that both operating systems and data management should be brought back into the languages. Ada can be viewed as the culmination of this drive.

A DBMS, ADAPLEX, is being developed for Ada. Documentation regarding ADAPLEX implies that Ada packages are being written in Ada to collectively perform the data management functions without violating the Ada groundrules. However, reading details leads to the conclusion that ADAPLEX will be a conventional DBMS written in Ada. This is unfortunate, particularly since ADAPLEX appears to be building a data language, DAPLEX, in Ada for ADAPLEX. It appears that Ada programs will not be able to access data held in ADAPLEX without using DAPLEX! This appears to be a violation of the spirit of Ada.(14)

In the full Ada Application support environment, equivalent capabilities can be provided inside the Ada Processing Support Environment without accepting the limitations now imposed by DBMS. These data handling capabilities can be provided by building a system of access 'engines' each of which corresponds to a known database system feature. These access engines should include a contiguous data group engine, a linear-list engine, a network navigation engine and a first-order logical (relational) engine. This will provide a framework

from which customized engines can be built to interface with the current DoD data files and DBMS, ensuring that current data can be accessed from an Ada environment without degrading that environment.

Each of these engines should be implemented as complete (closed) functions which might or might not make use of other engines. Each should also be run time support engines invoked by Ada programs as merely different devices or device types. There is no inherent limit to the number and kind of engines that an Ada program could 'open' during a run-time session. Application specific engines can be constructed in this framework such as an engine for exploring geometrical descriptions used in cartography, route search, etc required by navigation, mapping and intelligence analysis. As newer features such as decision aids become available, new engines can be constructed to represent these as directly as possible, probably using existing engines for support. Engines can be transitioned to hardware if desired, without disrupting the processing environment in which they reside. This approach ensures an open evolutionary information resource management capability inside the Ada APSE, whereas the traditional data model approach is a prescription for obsolescence.(15)

THE BUILDING OF THEORY

In the living of life, every mind must face the unyielding rock of reality, of a truth that does not bend .....To some men this will be an exultant challenge: that so much can be known and truth not be exhausted, that so much is still to be sought...To others this is a humiliation not to be borne; for it marks out sharply the limits of our proud minds.

St Thomas Acquinas, Summa Theologica

The new capabilities available when secondary storage was developed provided a great increase in capability, coupled with enormous additional complexity. This complexity was dealt with intellectually in many ways.

A DBMS holds data abstracted from some real situation, and is therefore a model of reality. This fact was recognized very early, as was the fundamental problem of mapping abstractions of reality into linear space, and the critical importance of being able to represent the inherent logical structure of the data.

The CODASYL committee developed it's model, which was intended more as a language with which to talk about all the new capabilities that had become possible than as something to be developed in it's totality.(16) They defined terminology which has been generally accepted; a data dictionary to carry information about data; and a data definition language and a data management language which are usually at least closely related and consistent in definition and usage, and so on. They defined the overall view of the data the enterprise seeks to manage as a schema, and individual subschema (now being referred to as viewpoints) of the individual activities to be supported by the data base.

The CODASYL committee intended to update their model as new capabilities become possible, maintaining a coherent 'state of data handling' as technology developed. To a large extent they have

succeeded in this goal. The DBMS as we know it can reasonably be called the child of CODASYL. However the very success in developing one coherent framework lead to monolithic integrated data management systems that became very difficult to adapt to changing technology.

Most of the early permanently valuable abstract theorizing regarding data handling from the intellectual ferment following the introduction of random accessible secondary storage appears to have originated in the IBM proprietary Universal Information Systems Technology (UIST) project, led by Dr Mike Senko, circa 1968-1971. Dr Senko, simply said, 'There are laws of how data behaves in storage media. We can and should codify these laws.' The Data Independent Accessing Methodology (DIAM) was developed by Dr Senko to provide a framework in which to report these laws. The UIST project was dissolved in 1971, and was reported in a the three-part article in the IBM Journal entitled 'Data Independent Accessing Methodology (DIAM)'. (17) Unfortunately, this article uses obscure notation and is extremely difficult to read. DIAM breaks information systems into layers of abstraction, with a formal model of each layer, and a formal description of the relationships between the layers. This turns out to be both powerful and necessary, and has become the standard procedure for discussing many different aspects of information resource management. Simulators built upon the DIAM theory have led to sufficient fidelity that the vendors of the system tested requested that the results not be published for proprietary reasons.

DIAM argues that information management systems are best described in terms of levels of functionality. The enterprise level defines the world-view from which the data is abstracted, and identifies semantic relationships between data. The information level identifies what

queries are possible and provides mechanisms to select from possible queries. The Access Path level identifies what access paths to the data have been implemented and provides mechanisms for selecting among implemented paths. The implemented DIAM simulator also allows identification of access paths which are implementable in the DBMS structure, but which have not yet been implemented. The encoding level identifies how each access path was or is to be implemented and selects from among implementation techniques. The address space level defines where each access path is implemented and selects from among logical files. And finally the physical device level identifies the device to which the path is assigned, selecting from among physical devices.(18)

The enterprise level requires a model capable of carrying semantic data, and some form of entity-set model is optimum. A semantic model is designed to capture and exploit the semantic similarities existing between different data models. The Entity-Category-Relationship model of data developed by Honeywell is an excellent basis for enterprise modeling.(19) Building an enterprise level model of the expanded enterprise facilitates using disparate DBMS together.(21) The information layer in DIAM and the access path layer are each best described in terms of the relational model. The relational model can be used without loss of generality to develop highly restrictive typing of access paths, and is the most powerful known model for syntactic interface and access path determination. The encoding level is system-specific since implementation techniques must be selected in terms of their efficiency. Both the address space and physical device level define specific relationships in the particular hardware

Another approach to the new capabilities to be achieved by secondary storage was developed by Dr Codd, also believed to have been a

member of the UIST. (21) In essence he said, 'the new capabilities are all very well, but what matters is how humans use the data and instead of optimizing usage I'll optimize the human interface and waste as much storage and processing power as necessary. People think in relations, I'll build a relational data base.' The result in practice was the implementation of a single data management system as a series of flat files, with complex software to perform comparatively simple operations upon them. This led to comparatively easy construction of queries. However, in practice there is no technical reason why the relational model should propagate beyond the user interface, and many reasons why more efficient software is developed when it does not. We are back to the problem of how to use our secondary storage.

Dr Codd and Dr Senko are known to have been at least in extreme competitive disagreement. This may well have retarded the recognition that their viewpoints need not conflict. In retrospect, much of the theorizing regarding DBMS during the 1970s appears to be extension of the dialogue between Drs Senko and Codd at the beginning of the decade.

Experience in using DBMS has established that the usual DBMS access is by application programs and not by individuals with ad hoc queries, so Dr Codd's arguments for the relational model lack generality. However, the well-designed relational user interface eases development of application programs, as well as facilitate end-user queries. In view of Dr Codd's rationale for introducing the relational model, it is ironic that one of the strongest arguments in favor of it is that a relational user interface facilitates restructuring a DBMS for efficiency of operation. It is a tribute to the pervasiveness of Dr Codd's arguments that the semantic models now being developed are referred to as post-relational.

The relational model is based upon the first order predicate calculus, which also provides a sound mathematical basis for primitive data handling procedures. Unfortunately the first order predicate calculus does not include such operations as count and sum, so at least a second order calculus needs to be developed. It is a serious theoretical limitation, and one reason why in practice mathematics isn't used in DBMS construction or usage.

Post-relational DBMS theorizing usually uses extended set theory. The relational model is subsumed under this theory. In practice, the predicate calculus appears to be appropriate to modeling the information interface in DBMS, and should be extended to cover the data situations of interest.

FOR A FIRM FOUNDATION

> Mind alone can and does discover heretofore unknown integral pattern concepts and generalized principals, apparently holding true throughout whole fields of experience. And once discovered by mind the concepts of the generalized principles become additional special-case experiences and are stored in the brain bank and retrievable thereafter by the brain. But brains and their externalized detachedly operating descendants--the electronic computer--can only search out and program the already experienced concepts, and mind alone can recognize and capture the unknown and unexpectedly existent, ergo, unsearchable, unwatched-for--generalized principles.
>
> Intuition, R. Buckminster Fuller, 1972

> At some point in our development of understanding about a subject, it becomes possible to state ways to determine facts and operations upon those facts, which always lead to other facts. At this point, the operations are called mathematics, and the determination of facts is called applied mathematics. The understanding about the subject can then be called science.
>
> Professor whose name I don't remember,
> circa 1948

To provide a sound mathematical basis for complete data handling, the set theory and relational algebra needs to be extended to at least a second-order calculus to deal with such operations as count and sum.

It also requires extension to deal with questions of completeness and relevance such as 'What data is known, unknown, not available, or inconsistent with other data?' and 'What data is relevant to this query? Is it more or less relevant than other data? If certain data is relevant to a query, what other data is still needed to answer the query and how can this need be expressed?'

Relational algebra can be extended to deal with these questions by extending or interpreting the notion of domain to give rigorous definition to the concepts of unknown, incomplete, inconsistent and not applicable; adding operators to the algebra and extending the domain definition of existing ones; discovering the laws for the manipulation of the extended algebra and developing algorithms for manipulating the algebra, particularly for the purpose of computing completeness and relevance. Such sucessful extension of relational algebra would lead to the ability to state, formally, what data is or is not relevant to a given situation so that inferences can then be made. (22)

The immediate payoff would be in direct support to information analysts either using the algebra directly or through application programs.

In the longer term this extended relational algebra could provide a badly needed rigorous mathematical foundation for development of artificial intelligence techniques. Extended relational algebra will also facilitate interfacing artificial intelligence with data held in data bases of any size and characteristics since the information level of any data base is well defined in terms of relational algebra.

Another approach to interfacing artificial intelligence programs with data held in DBMS has been developed by SDC.(23) They have defined a semantic model patterned after the AI 'concept net' approach and demonstrated isomorphism between that and a DBMS model they also developed. They expect this to lead to implementation of a simple model which will simultaneously serve as a knowledge base for an expert system and a schema for data base applications.

Artificial Intelligence, Decision Aids and Knowledge Management bring us full circle. They are unavoidably dependent upon values, whether or not these values are explicitly stated. They will be of maximum value only when they rest upon sound theory. Until then, they are at best 'ad hockery.'(24)

One of the most difficult aspects of theory to discuss is data representation. A data base represents some aspects of reality in symbols that can be held in computers. Some representation problems are long solved and readily agreed upon; time is divided into days, hours, minutes, etc, and the only decision is which of a small number of representations shall be chosen for a specific system. Sometimes the representation chosen interacts seriously with new uses for the data. For example, the decision to manage data in terms of fixed installations makes dealing with mobile objects almost intractable. We don't know how to deal with multiple representations of the same object. Attempts to address this problem lead to fundamental philosophical problems, and philosophers have not yet come to terms with the need to solve very mundane examples of their very abstract theorizing.

As previously noted, we need the capability to identify and track more than one concept at a time. We need to recognize that 'it is'

always in support of more than one concept. We also urgently need theory that we trust enough so that when concepts are proven incorrect we accept the proof. Then we need methodology to deal with the disproven concepts. Sometimes we will be able to use the proof that our concepts are incorrect to develop better concepts. Sometimes we will only know that what we thought was correct is wrong, without enough knowledge to identify what is right. This is the most difficult of all intellectual situations. We have to be able to accept the fact that we don't know. We must also find a way to describe what is happening well enough to bring brains to bear on the newly identified missing concepts. The methodology we require must be a validated theory about knowledge. It must also be representable in digital processors if it is to be a viable foundation for information resource management.

One promising approach to developing this methodology is the state-of-affairs descriptive technology developed by Dr Peter Ossorrio at the University of Colorado at Boulder under the rubric Descriptive Psychology.

State of Affairs Technology (25)(26)(27) maintains that information analysis is at once both completely principled yet completely context-dependent. To deal effectively with these a number of techniques have been developed. These techniques include Paradigm Case Formulation as opposed to 'definition'(28); Judgement-Space as opposed to 'identification'(29); State of Affairs Systems as opposed to 'frames'(30); Intentional Action Systems as opposed to 'scripts'(31); and Ex Post Facto Formulation as opposed to 'time'(32). These techniques have been successfully applied to solving problems in expert systems (33), linguistic data processing (34), and automatic fact analysis(25). The evidence of these successes suggest that the

state-of-affairs technology is a fruitful basis for research in representation, and the development and validation of concepts.

Until we have agreed-upon formalisms by which we can manage our conceptual models of reality, capable of representing multiple world-views we do not have valid theory. Until then, our application programs, decision aids, knowledge management and artificial intellignce programs can only be validated in experiential terms. We will have no reliable procedures to be sure they apply in new situations, and new situations are precisely where they will be most needed.

REFERENCES

1. Vernimb, Carl O "The European Network for Scientific Technical and Social Information" Nachrichten fur Dokumentation 28. Jahrgang Nr 1-Feb 1977

2. Personal conversation with Robert Walker, RADC July 1983

3. McGee, W.C., "A Contribution to the Study of Data Equivalence" in Data Base Management, Klimbie, J.W. Koffman, K.L. eds. North Holland N.Y. 1974

4. Tsichritzis, D.C. Klug, A. eds. The ANSI/X3/SPARC DBMS Framework Report on the Study Group on Database Management Systems

5. NBSIR 82-2619 Functional Specifications for A Federal Information Processing Standard Data Dictionary System

6. Schmidt, Joachim W Brodie Michael L, eds. Relational Database Systems, Analysis Comparison, Springer-Verlag Ny 1983

7. Shoshani, Ari I Spiegler, "The Integration of Data Management Systems on a Computer Network" AIAA Computer Network Systems Conference, Huntsville, AL April 16-18 1973

8. Frankel, Robert E Buneman, O.P. FQL-A Functional Data Base Query Language Proceedinngs of the ACM-SIGMOD 1979

9. Savas, Mary Ann personal conversation, Oct 1892

10. Schneider, L.S. "A Relational Query Compiler for Distributed Heterogeneous Data Bases" SHARE 50 Proceedings, Denver CO March 1978

11. Shoe salesmen in Longmont CO and Rome NY told me this as fact.

12. Bouille, Technical Presentation RADC, Circa 1979

13. Ada is a registered trademark of the Department of Defense

14. Shipman, D. "The Functional Data Model and the Data Language DAPLEX" ACM Transactions on Database Systems, Vol. 6, No 1, Mar 81

15. I owe this concept to stimulating ARPANET exchanges with Lowell Schneider of Denver University. Kirk Sattley of Mass. Computer Associates also helped me focus my opinion.

16. ---- Feature Analysis of Generalized Data Base Management Systems, The CODASYL Systems Committee Technical Report, May 1971

17. Senko M.E. "Data Independent Architecture Model--Four Levels of Description" Report RJ982 IBM Research, Yorktown Hts., N.Y

18. Levin M "An Introduction to DIAM-Levels of Abstraction in Accessing Information" ACM 1978 Wash DC

19. Weeldreyer, J.A. The Entity-Category-Relationship Model of Data, Technical Report No. HR-80-250 Honeywell CCSC, Bloomington Minn Dec 80

20. Schneider, Lowell Levin, Michael "Data Independent Accessing Methodology Distributed Access" RADC-TR-81-137 June 1981

21. Codd, E.F. "A Relational Model of Data for Large Shared Data Banks" Comm. of the ACM Vol 13 No 6 June 1970

22. I owe this approach to Enrique Ruspine, then of Logicon. Michael Levin, then of Denver University extended and developed the concept in further discussion.

23. Personal discussion with Iris Kameny, Workshop on Data Base Machines, Blue Mountain Lake July 5-8 1983

24. This striking phrase originated with Dr Jack Minker, U of Maryland

25. Ossorio, P.G., "Classification Space: A Multivariate Procedure for Automatic Document Indexing and Retrieval," in Multivariate Behavioral Research, Vol. 1, October 1966.

26. Ossorio, P.G., "What Actually Happens", University of South Carolina Press, Columbia SC, 1978.

27. Ossorio, P.G., Attribute Space Development and Evaluation, Rome Air Development Center Technical Report RADC-TR-67-640, Rome Air Development Center, Rome NY, 1968.

28. Ossorio, P.G., "Paradigm Case Formulation: The PCF and Related Types." in Davis, K.E. (ed.), Advances in Descriptive Psychology, Vol.1. JAI Press, Greenwich CT, 1981

29. Jeffrey, H. J. "A New Paradigm for Artificial Intelligence" in Davis, K. E. (ed) Advances in Descriptive Psychology, Vol. I, JAI Press, Greenwich CT, 1981.

30. Ossorio, P.G., State of Affairs Systems: Theory and Techniques for Automatic Fact Analysis, Rome Air Development Center Technical Report RADC-TR-71-102, Rome Air Development Center, Rome NY, 1971.

31. Ossorio, P.G., "Outline of Descriptive Psychology," in Davis, K.E. (ed.), Advances in Descriptive Psychology, Vol. 1, JAI Press, Greenwich CT, 1981

32. Ossorio, P.G., "Ex Post Facto Formulation," in Davis, K.E. (ed.), Advances in Descriptive Psychology, Vol. 2, JAI Press, Greenwich CT, 1982

33. Jeffrey, H.J. Puttman, A.O., "The MENTOR Project: Replicating the Functioning of an Organization," in Davis, K.E. (ed.), Advances in Descriptive Psychology, Vol. 1, JAI Press, Greenwich CT, 1981

34. Johannes, J.D., "Automatic Thyroid Diagnosis via Simulation of Physician Judgement," Doctoral Dissertation, Vanderbuilt University, 1977 (University Microfilms No. 77-30,364).

MISSION
of
Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control Communications and Intelligence (C^3I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

LMED

8